C Switch Statement

The switch statement in C is an alternate to if-else-if ladder statement which allows us to execute multiple operations for the different possibles values of a single variable called switch variable. Here, We can define various statements in the multiple cases for the different values of a single variable.

The syntax of switch statement in c language is given below:

```
1. switch(expression){
2. case value1:
3.  //code to be executed;
4.  break;  //optional
5. case value2:
6.  //code to be executed;
7.  break;  //optional
8. ......
9.
10. default:
11.  code to be executed if all cases are not matched;
12. }
```

Rules for switch statement in C language

1) The *switch expression* must be of an integer or character type.

2) The *case value* must be an integer or character constant.

3) The *case value* can be used only inside the switch statement.

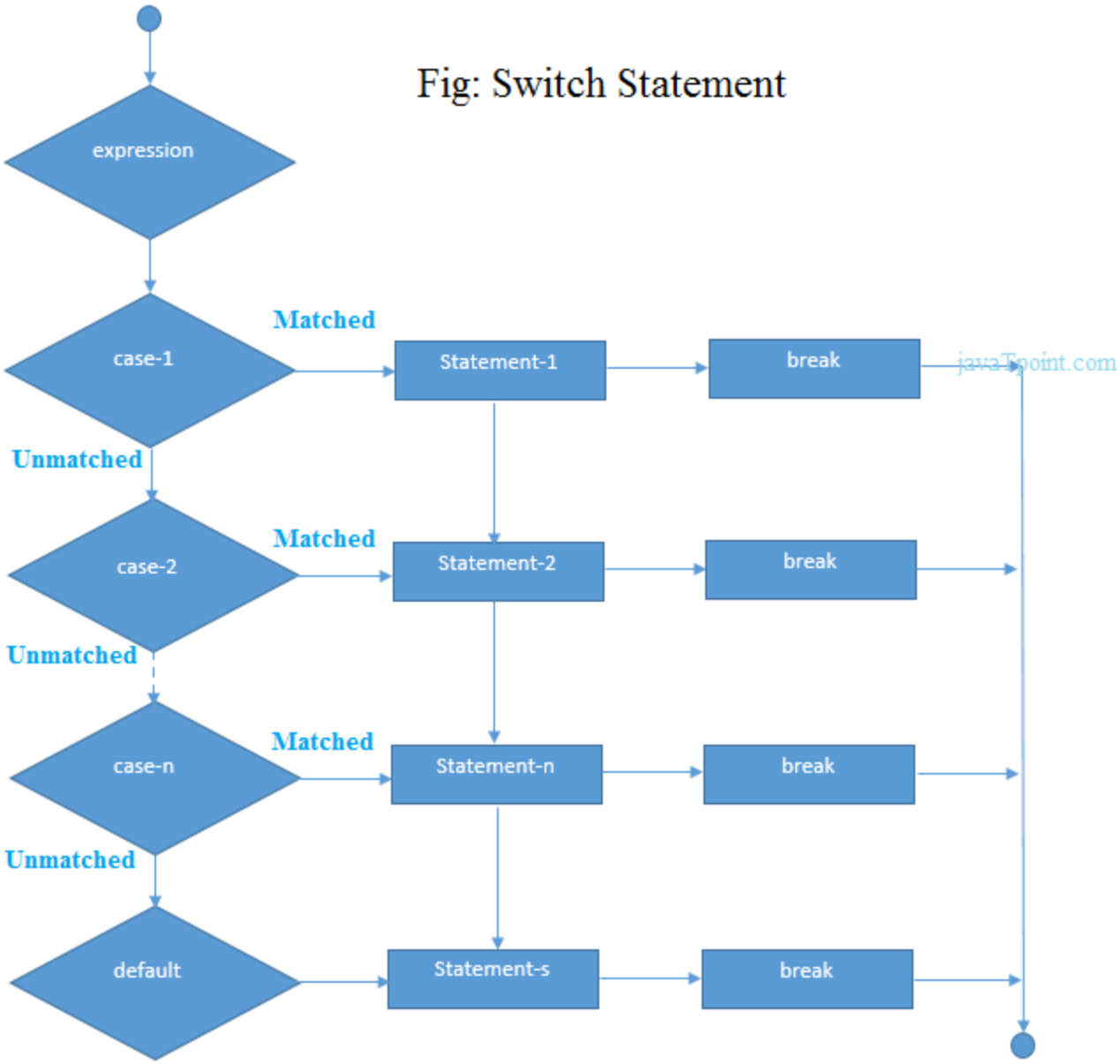
4) The *break statement* in switch case is not must. It is optional. If there is no break statement found in the case, all the cases will be executed present after the matched case. It is known as *fall through* the state of C switch statement.

Let's try to understand it by the examples. We are assuming that there are following variables.

```
1. int x,y,z;
2. char a,b;
3. float f;
```

Valid Switch	Invalid Switch	Valid Case	Invalid Ca
switch(x)	switch(f)	case 3;	case 2.5;
switch(x>y)	switch(x+2.5)	case 'a';	case x;
switch(a+b-2)		case 1+2;	case x+2;
switch(func(x,y))		case 'x'>'y';	case 1,2,3;

Flowchart of switch statement in C



Functioning of switch case statement

First, the integer expression specified in the switch statement is evaluated. This value is then matched one by one with the constant values given in the different cases. If a match is found, then all the statements specified in that case are executed along with the all the cases present after that case including the default statement. No two cases can have similar values. If the matched case contains a break statement, then all the cases present after that will be skipped, and the control comes out of the switch. Otherwise, all the cases following the matched case will be executed.

Let's see a simple example of c language switch statement.

```
#include<stdio.h>
int main(){
int number=0;
printf("enter a number:");
scanf("%d",&number);
switch(number){
case 10:
printf("number is equals to 10");
break;
case 50:
printf("number is equal to 50");
break;
case 100:
printf("number is equal to 100");
break;
default:
printf("number is not equal to 10, 50 or 100");
}
return 0;
}
```

Output

```
enter a number:4
number is not equal to 10, 50 or 100
enter a number:50
number is equal to 50
```

Switch case example 2

```
#include <stdio.h>
int main()
{
    int x = 10, y = 5;
```

```
    switch(x>y && x+y>0)
    {
        case 1:
        printf("hi");
        break;
        case 0:
        printf("bye");
        break;
        default:
        printf(" Hello bye ");
    }

}
```

Output

```
hi
```

C Switch statement is fall-through

In C language, the switch statement is fall through; it means if you don't use a break statement in the switch case, all the cases after the matching case will be executed.

Let's try to understand the fall through state of switch statement by the example given below.

```
#include<stdio.h>
int main(){
int number=0;

printf("enter a number:");
scanf("%d",&number);

switch(number){
case 10:
printf("number is equal to 10\n");
case 50:
```

```
12. printf("number is equal to 50\n");
13. case 100:
14. printf("number is equal to 100\n");
15. default:
16. printf("number is not equal to 10, 50 or 100");
17. }
18. return 0;
19. }
```

Output

```
enter a number:10
number is equal to 10
number is equal to 50
number is equal to 100
number is not equal to 10, 50 or 100
```

Output

```
enter a number:50
number is equal to 50
number is equal to 100
number is not equal to 10, 50 or 100
```

Nested switch case statement

We can use as many switch statement as we want inside a switch statement. Such type of statements is called nested switch case statements. Consider the following example.

```
1.  #include <stdio.h>
2.  int main () {
3.
4.     int i = 10;
5.     int j = 20;
6.
7.     switch(i) {
8.
9.        case 10:
10.          printf("the value of i evaluated in outer switch: %d\n",i);
11.       case 20:
```

```
12.         switch(j) {
13.            case 20:
14.               printf("The value of j evaluated in nested switch: %d\n",j);
15.         }
16.   }
17.
18.   printf("Exact value of i is : %d\n", i );
19.   printf("Exact value of j is : %d\n", j );
20.
21.   return 0;
22. }
```

Output

```
the value of i evaluated in outer switch: 10
The value of j evaluated in nested switch: 20
Exact value of i is : 10
Exact value of j is : 20
```